# Capstone Project Proposal: Extending Virtual Memory Management in Xinu

Thomas Gagne

## 1 Project Description

For this project I plan to implement a virtual memory manager using demand paging for the Xinu operating system. Currently, Xinu has support for elementary 1-1 paging support for processes and uses these pages to provide virtualized separate address spaces. Xinu additionally has support for MMUs and TLBs if they are present. Hence, the primary goal of the project will be to extend this current paging system by implementing demand paging which utilizes a swap space on disk. Implementing this will involve several general steps: First, it will be necessary to partition the disk to include a swap space in such a way that Xinu can recognize the swap space. Second, it will be necessary to modify the current structure of the paging system (i.e. PTEs) to recognize invalid pages and to include the necessary information for a page replacement algorithm. Third, it will be necessary to create a handler which performs the task of page swapping through a page replacement algorithm. And finally, it will be necessary to combine each of these components by modifying the current memory management handlers to recognize invalid pages and call upon the new handler as necessary.

While this project's focus is not necessarily novel since the topic of virtual memory management is a well-studied subject, this project will be a substantial extension to Xinu's current memory management system due to it involving modifying and combining the existing functionality of numerous components in Xinu, namely the various handlers for granting, relinquishing, and accessing memory for threads as well as the utilities for disk access and partitioning. I will therefore need a complete understanding of each of these components so that I can seamlessly combine them to form a functional final project. This is where the main benefit of the project lies: performing comprehensive research into the inner workings of a pre-existing operating system, using that information to develop an appropriate plan for extending the operating system, and using this plan to guide implementation of this extension. In particular, this will be an opportunity to take the concepts covered in the Operating Systems course which were taught from a largely theoretical standpoint and determine how to most effectively apply those concepts to shape the final project given the structure of Xinu's provided components. This will give me valuable experience in understanding operating systems development and design as well as experience in a project involving a considerable amount of research and using that research to guide implementation.

As resources for this project I will use my experience in Xinu development from the Operating System course as well as *Operating System Design: The Xinu Approach* by Douglas Comer and the numerous sources online covering Xinu's details in order to gain an understanding of the components that will be involved in this project. To perform development, I will use a cross-compiler as described at `http://xinu.mscs.mu.edu/Build_Xinu` to compile Xinu to MIPS and I will use Qemu-mipsel to virtualize Xinu in a MIPS environment.

I anticipate that the primary point of uncertainty in being able to complete this project will be having Xinu recognize and utilize a portion of the disk for swapping space. To reduce this uncertainty, I plan on focusing on this task early on in the project to determine whether or not implementing this component will be feasible. If I find that this will not be possible to complete in a reasonable amount of time, I instead plan to implement a simulated swap space in a dedicated portion of memory, which can either be done by reserving a portion of user or kernel space upon initialization or by modifying the currently defined regions for user and kernel space to include a third space for swapping. To additionally ensure the completion of as much of the project as possible, I plan on tiering the implementation of components in such a way that a primitive swapping mechanism is completed early on and can be used as the foundation for a more complex system. If towards the end of the semester I find myself unable to complete implementation in the time given I will instead focus my efforts on a comprehensive paper summarizing the current model for memory management in Xinu and describing my proposed model. Given my plan of progressively implementing the necessary components, however, I think it is highly unlikely that this would become necessary.

## 2 Tentative Project Weekly Breakdown

| Week | Summary |
|---|---|
| 1 | Establish development environment by testing that the cross-compiler works properly and that the compiled result can be virtualized appropriately. |
| 2 | Decide whether or not I will pursue using a portion of the disk for swap space or if it will be necessary to use a simulated virtual swap space. |
| 3 | Based upon the decision made in week 2, reserve a dedicated portion of either disk memory or virtual memory to be used as swap space. In the latter case, this can be done by either modifying the memory region definitions in the include/mips.h file or by modifying the memory allocation handlers to not allocate memory in a given segment. |
| 4 | Finalize work on reserving a dedicated swap space. |
| 5 | Begin studying the current memory management handlers for allocating, accessing, and relinquishing memory to begin designing their extensions. Respectively, this involves designing a plan for: checking that the page can be placed into memory and if not, swap out a page to swap space; checking that the requested page is valid and if not, load it in from swap space; modifying the PTEs to mark the page as free memory, whether it is in memory or in swap space. |
| 6 | Continue studying the memory handlers to finalize designs for their extensions. |
| 7 | Begin implementing the designed extensions for the memory handlers. At the core of many of these components is the page replacement algorithm, and early on this algorithm will be as simple as possible. |
| 8 | Continue implementing extensions for the memory handlers. |
| 9 | Continue implementing extensions for the memory handlers. |
| 10 | Finalize implementing extensions for the memory handlers and begin research into more complex page replacement algorithms |
| 11 | Continue research into an appropriate page replacement algorithm. |
| 12 | Begin implementing the chosen page replacement algorithm, which will involve changes to the structure of PTEs as well as modifications to the extended memory management handlers to incorporate the new page replacement algorithm. |
| 13 | Continue implementing the page replacement algorithm. |
| 14 | Finalize implementing the page replacement algorithm and focus on resolving any outstanding issues. If time permits, begin looking into experimental ways to improve upon the page replacement algorithm (e.g. preemptive swapping out old and unused pages). |
| 15 | Finalize project and if time permits, continue research into experimental improvements upon the page replacement algorithm. |